Histogram-based Object Tracking

Meghna P Ayyar and Smriti Joshi

I. INTRODUCTION

This laboratory report presents histogram based single object tracking routines, written in C++ using the OpenCV 4.2.0 library. The development platform used are: Linux with the Eclipse IDE and Windows using the Visual Studio Community 2019 IDE. The implementation is based on three modules: color based, gradient based and a fusion of the color and gradient based tracking. The results have been evaluated on the video sequences given in Table I and are summarized in Section V.

II. METHOD

The current task focuses on using intensity based histograms to track a single object in a given video sequence. Three methods for the calculation of the intensity gradients have been implemented namely: color based, gradient based and a fusion of colour and gradient based intensities as given by [1]. Further details regarding the implementation have been summarized in Section III.

A. *Colour-based tracking*

Color based tracking uses normalized histogram of color features for the representation of the object that is being tracked from frame to frame. Using either a single or multi colour channel histogram, a target representation, also called the model histogram, is computed for the object before the tracking begins. At run time, in every frame, a set of candidates are generated based on predetermined conditions. The normalized histogram representation is then computed for each of these generated candidates in the same way the model histogram was calculated. A distance metric is then used to find the best fit (most similar) to the model histogram from the possible candidates generated. Currently the Bhattacharya distance (1) is used for the distance metric. It is computed for every pair of model histogram and the candidate histograms. The candidate with the closet representation to the model histogram (has the minimum Bhattacharya distance) is then chosen as the best detection of the object in the current frame.

$$D(p,q) = -\ln(BC(p,q)) \tag{1}$$

where, *D(p,x)* is the Bhattacharya distance and *BC* is the Bhattacharya Coefficient given by (2), In this case, *p* is the normalized model histogram and *q* is the normalized candidate histogram .

$$BC(p,q) = \sum_{x \epsilon X} \sqrt{p(x)q(x)} \tag{2}$$

B. *Gradient-based tracking*

Gradient based tracking uses the gradient information of the object to create the model representation. The gradient features computed by the authors of [1] are based on the normalized sum of gradients around the perimeters pixels of the region under consideration. This region could be a bounding box or a bounding ellipse. Currently, the objects being tracked are denoted by bounding boxes. Except for the difference in the features generated to compare the model and the generated candidates, the gradient based tracking pipeline is very similar to that of the color based tracking. For comparing the model and the candidate histograms the L2 distance (3) is used instead of the Bhattacharya distance. The candidate closest to the model representation (minimum L2 distance) is then chosen as the best detection of the object in the current frame.

$$d(p,q) = \sqrt{\sum_{i=1}^{n} (qi - pi)^2} \tag{3}$$

where, *d(p,q)* refers to the L2 distance, *n* is the number of gradient features calculated for one region, pi is the normalized gradient feature for the model and qi is the normalized gradient feature of the candidate.

C. *Fusion based tracking*

Color based trackers and gradient based trackers compute different set of features for tracking which makes them invariant to different things and hence their performances also vary. Color based trackers do not consider the geometry of the object and hence work well in cases where objects can shape differences (non-rigid body). In contrast they would have challenges in discriminating objects that have similar color but a gradient based tracker may be complementary in that case. This complementary behaviour of the color and gradient based trackers was observed by the authors of [1], which has been implemented and tested in current work. By combining the results of the two trackers, a final candidate is selected that minimizes the overall error of this combination, as the best detection of the object in the current frame. Further performance details of this tracker have been summarized in Section V.

III. IMPLEMENTATION

This section explains in detail the implementation of the methods described in section II. It includes the inputs, outputs and other coding considerations followed to achieve the objectives.

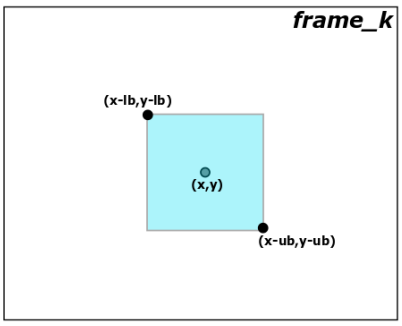

Fig. 1. Candidate generation neighbourhood. (x,y) - center pixel of detected object in $frame_{k-1}$. Blue region is the search area for new centers

A. *Candidate generation*

The function gen_candidates() implements the candidate generation algorithm for each of the three tracking methods mentioned in Section II. It follows the grid algorithm proposed in [2] as shown in Fig 1. The candidates for the current frame $frame_k$ are generated from an NxN neighbourhood, with respect to the center of the previous detection in $frame_{k-1}$. In the current implementation, the height and width of the generated candidate is kept fixed to the height and width of the initial bounding box used to create the model histogram. The function has two major parameters namely, stride - the stride for the neighbourhood and numCandiates - the total number of candidates to be generated. Based on numCandidates parameter the function calculates the value N for the neighbourhood grid as given by (4).

$$N = \sqrt{numCandidates} \tag{4}$$

Given the center coordinates of the detection at $frame_{k-1}$, the lower and upper bounds of the NxN grid around the center coordinates for the current frame $frame_k$ is calculated as shown in (5) and (6), where lb stands for lower bound and ub stands for upper bound.

$$lb = \frac{N}{2} * stride \tag{5}$$

$$ub = \begin{cases} (\frac{N}{2} - 1) * stride, & \text{if } x \mod 2 == 0 \\ \frac{N}{2} * stride, & \text{otherwise} \end{cases} \tag{6}$$

For an even number of candidates the search condition varies a little and hence the two cases have been included while calculating the upper bound. The lower bound and upper bound provide the search area for new candidate centers around the center coordinate of the previous detection. Iterating through each with a step of stride, N^2 candidates are generated.

B. *Color based tracking*

Color based tracking is implemented by the ColorTracker class. The tracker implements a single channel histogram calculated based on one of the seven possible features: Grayscale, Red, Green, Blue, Hue, Saturation or Value that are extracted from the region of interest. The model histogram corresponds to the histogram calculated from ground truth bounding box of the first frame of the video sequence. The main function creates and initializes a ColorTracker object with the parameters: feature, stride, number of bins for the histogram and the number of candidates regions to be generated. The function start() of the ColorTracker object receives as argument the current frame and performs the calculations for tracking. It then returns the best fit candidate as Rect() object. In addition, the ColorTracker object also contains the model histogram as a member which is computed and stored when the first frame is passed to the start() function. The specified candidates are generated by the get_candidates() function that returns a vector containing Rect() objects for each of the candidates. The class also contains get_rect() and get_center() functions to calculate the left corner points given the center of the box and vice versa.

Based on the feature parameter given to the object, the get_histogram() function computes the corresponding histogram with the given number of bins for each candidate. The histograms are then normalized using the normalize() of OpenCV with an L1-norm. Iterating through the vector of calculated histograms, the Bhattacharya distance for the candidate histogram and model histogram are computed using the calc_distantce() function. The scores are accumulated in vector called as candidate_scores. The class also has two getter functions of get_neighbours() and get_scores() that return the candidate list and candidate scores respectively. The index of the minimum element of the candidate_scores vector corresponds to the best detection for the object in the current frame. The function returns the corresponding candidate to the main function which is then compared to the ground truth to check for the accuracy of the match.

C. *Gradient based tracking*

Gradient based tracking is implemented by the GradientTracker class. The basic structure and flow of computation is similar to that of the ColorTracker class mentioned in the section III-B. It differs from the ColorTracker as it computes gradient features and uses the L2 distance for scoring. Instead of calculating gradients at the perimeter of the bounding box, HOG descriptors are extracted for the whole region of interest. The HOG descriptor is initialized with the default parameters and the number of bins can be changed as a parameter when the object is created. The model features are computed from the ground truth bounding box from the first frame in the start() function. The candidates are generated in the same way as the ColorTracker class. The class also contains get_rect() and get_center() functions to calculate the left corner points given the center of the box and vice versa.

Before computing the HOG descriptor, each candidate region is resized to match the dimension HOG descriptor. The descriptor is then computed based on the bins parameter provided and normalized using the normalize() function and L1 norm. The calc_distance() function is used to calculate the L2 distance between each of the model

descriptor and the candidate descriptors and is stored in the candidates_scores vector. The class also has two getter functions of get_neighbours() and get_scores() that return the candidate list and candidate scores respectively. The index of the minimum element of the candidate_scores vector corresponds to the best detection for the object in the current frame. The function returns the corresponding candidate to the main function which is then compared to the ground truth to check for the accuracy of the match.

D. Fusion based tracking

Fusion based tracking is implemented by the FusionTracker class. The class contains one object each of the ColorTracker and the GradientTracker class as its members. The FusionTracker object from the main function receives as parameters the mode of execution, feature, stride, bins and number of candidates. The class implements three modes of execution: only color based tracking, only gradient based tracking and the fusion of color and gradient. For the only color and only gradient the corresponding object is initialized by the FusionTracker object and the correspoding start() functions are called to do the necessary computations. The flow of computation for these two modes follow the ones mentioned in their corresponding implementations.

In the fusion mode, the combine_results() function performs the necessary computations. The function calls the start() of both the ColorTracker object and the GradientTracker object and passes the current frame as the argument to both the functions. As both the objects implement the same get_candidates() algorithm, the getter funciton get_neighbours() of one of the object is used to create the list of the generated candidates. The get_norm_scores() getter function is invoked for both the objects and the normalized scores for both the trackers are saved in vectors cscores and gscores of the combine_results() function. Using the transform() function of C++ these two vectors are added and the index of the resulting vector with the minimum value is computed. The candidate corresponding to this index is then chosen as the match for the tracked object in the current frame and is returned to the main function. By combining the normalized scores, the idea is to choose the candidate that minimizes the distance in both color and gradient features for the object model.

E. Running the code

Three separate source folders have been submitted for each of the trackers implemented.

- **ColorBasedTracker** : To run color based tracker change the parameters given in the main.cpp in the ColorBasedTracker folder. feature, stride, bins and numCandidates can be set in the main function. The exact feature code to select one of the 7 features is mentioned in the document. The default parameters are, feature: Saturation, stride: 2, bins:16 and numCandidates:121. Press escape to view the similarity plot and move to the next video.

Task	Video	Description
Color based Tracking	bolt1	Runners look similar Similar skin color Similar color jerseys
	sphere	Low resolution Reflective surface so varying color
	car1	Shaky camera, backview similar to nearby vehicles
Gradient based Tracking	bolt1	Runners have similar structure, Variable poses
	ball2	White ball on similar color background, fast movement of object
	basketball	Similar appearance, fast movement halfway in video
Fusion based tracking	bag	White polythene, random motion, variable size of bounding box
	ball	Variable size of object due to diagonal movement, large background included in box
	road	Low resolution, trees occlude object sometimes

TABLE I
DATASET DETAILS

- **GradientBasedTracker** : To run the gradient based tracker, set the parameters in main.cpp. By default they have some values set. Stride=2, bins=9 and numCandidates=100 and can be changed in the main function. Press escape to view the similarity plot and move to next video.
- **FusionTracker** : To run the fusion based tracker change the parameters of the main.cpp for the FusionTracker folder. The file offers three modes of execution: 'C' corresponds to color only, 'G' corresponds to gradient only and 'F' corresponds to the fusion mode. Other parameters like features, stride, bins and numCandidates for the two trackers can also be set from the main function. The default parameters are feature: Gray, stride: 2, bins:9 and numCandidates:100. Press escape to view the similarity plot and move to next video.

IV. DATA

The dataset used for evaluation consists of nine sequences, each with different critical factors for evaluation. The set of videos chosen have different types of challenges that are summarized in Table.I

V. RESULTS

The results for three types of tracking implemented have been summarized in this section. All the figures provided for the analysis have the green bounding box as the ground truth and the red bounding box as the detection by the tracker.

A. Color based tracking

The analysis of the color based tracking has been performed for three sequences, each with their own challenges.

Channel	numCand	Accuracy	Time(ms/frame)
Red	121	0.14	1.42
	81	no res	no res
Green	121	**0.22**	1.46
	81	0.13	**1.06**
Blue	121	0.14	1.25
	81	0.13	1.29
Gray	121	no res	no res
	81	no res	no res
Hue	121	0.21	3.18
	81	0.21	2.63
Saturation	121	0.18	3.45
	81	0.17	2.19
Value	121	0.16	3.55
	81	0.16	2.84

TABLE II

COLOR BASED TRACKING RESULTS FOR BOLT1. NUMCAND: NUMBER OF CANDIDATES

Color based tracking offers seven features: Gray Channel, Red Channel, Green Channel, Blue Channel, Hue Channel, Saturation Channel and Value Channel. The following sections discuss the performance of these features for the each of the sequences.

1) ***bolt1****:* This video is used to establish the correctness of the code implementation. As summarised in Section IV, a few challenges presented by the video are the similarity of the object of interest with other objects in the scene. Also, after frame 170, the camera angle changes. This results in change of scale of the object. This is partly handled by the normalised histograms. But since the bounding box size and neighbourhood size are fixed and location update is not fast enough, there is a drift observed in the bounding box towards other objects/regions after a certain frame. This video is evaluated for the suggested parameters of bins=16, number of candidates=121 and stride=2. The features used are: red, green, hue, gray. Value, saturation and blue channel are not expected to show differentiating characteristics.

Fig. 2. Results of the track analysis from red channel for different frames (bolt1, from table 2).

a) Red Channel: The object of interest is a runner of black skin color, and, yellow-green jersey. Due to this colour combination some differentiability is expected in the Red Channel. From frames 1-150, maximum overlap with the ground truth bounding box is seen. The average processing time is 1.30 ms/frame and the average tracking performance is 14.1% accuracy. This can be visualised from Figure 2 which shows current frame with detection at different instances. It can be observed that at frame 158, the bounding box starts drifting as the object size starts changing due to the viewing angle. Figure 3 shows the comparison between the histograms of good and bad detections. It is observed that the histograms have similar shape and peak around the same region. The peak is slightly higher for runner with red jersey whose bottom half only is detected. This points to the drawback of histogram as a way of target representation as it just takes into account the distribution of pixels and not the geometry of the location or the shape.

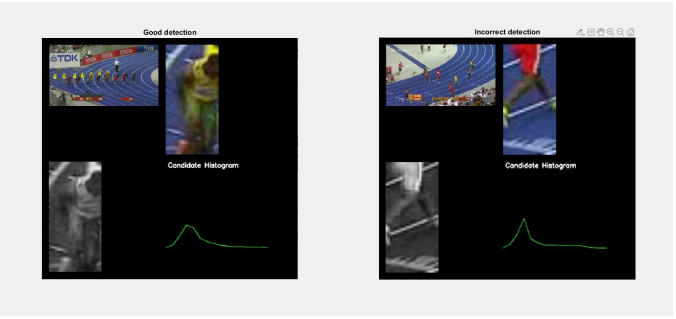

Fig. 3. Comparison of good and bad detection from red channel(bolt1, from table 2)—Right:Good detection, Left:Bad detection - Top-left:Current Frame, Top-right: Candidate Region, Bottom-Left: Feature, Bottom-right: Histogram of Candidate Region

b) Green Channel: In this channel, the best performance is observed as the ground truth overlaps with the detected bounding box from frame 1 to frame 200. This is probably due to the clear dominance of green and yellow colors in the object of interest. However, the accuracy scores are not satisfactory enough. The average processing time is 2.03 ms/frame and the average tracking performance is 22.4%. Figure 4 shows the comparison of good and bad detections for this channel. It can be observed that the histograms peak at different positions. Even in that case the candidate has a maximum similarity to the histogram because at this instance, the bounding box drifts far away and the object of interest of observable only on a smaller scale.

c) Hue Channel: Figure 5 shows the results observed for the hue channel for the given parameters. The object is detected correctly up to frame 170. The average processing time is 4.09 ms/frame and average tracking performance is 21.1%. Compared to the green channel, it takes more time and gives about the same accuracy. It is observed that when the bounding box drifts, it moves towards another object with yellow shirt, green shorts and dark skin. The histograms for the two objects are very similar as seen in Figure 6. Unlike other channels, it can be said that this channel

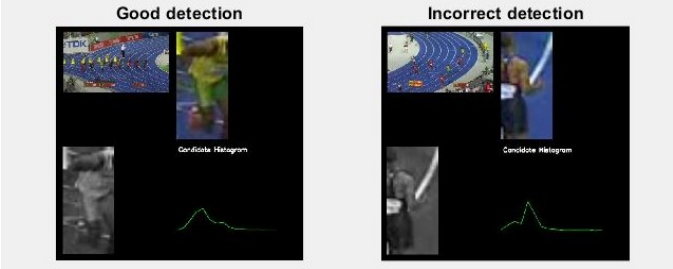

Fig. 4. Comparison of good and bad detections from green channel(bolt1, from Table 2)—Right:Good detection, Left:Bad detection - Top-left:Current Frame, Top-right: Candidate Region, Bottom-Left: Feature, Bottom-right: Histogram of Candidate Region

Fig. 5. Results of the track analysis from hue channel for different frames (bolt1, from table 2).

offers better discrimination between objects with differently colored jerseys as the object moves towards another highly similar object.

d) Gray Channel: Gray channel gives comparatively worse results than the other channels as the information is not discriminative enough. Figure 7 shows the histograms observed from three different objects using the gray features. They are highly similar to the model histogram.

Figure 8 shows the similarity maps obtained by comparing the detected bounding boxes with the ground truth. As it can be clearly seen, the green channel gives higher scores for

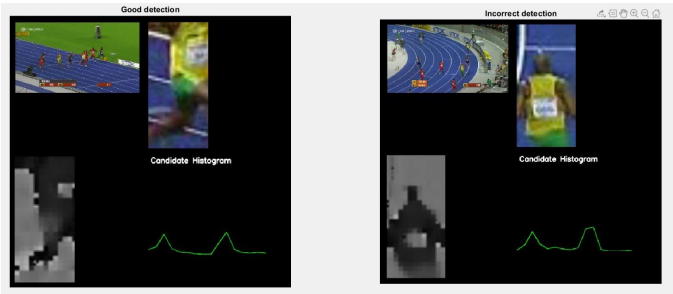

Fig. 6. Comparison of good and bad detection from hue channel(bolt1, from table 2)—Right:Good detection, Left:Bad detection - Top-left:Current Frame, Top-right: Candidate Region, Bottom-Left: Feature, Bottom-right: Histogram of Candidate Region

similarity and find relevant bounding boxes for frames later in the sequence as well. Table II gives the scores for the sequence. The best set of parameters for this video is green channel with 16 bins and 121 candidates.

*2) **Sphere**:* As summarized in Section IV, a few challenges are presented by this video for color based tracking. The frames are low resolution and the object of interest is visible through a printed glass. Therefore, the appearance of the object changes through the sequence. The person rotates the ball around its axis which results in his hand covering the ball. The motion of the ball is also random. This video is evaluated for the suggested parameters of bins=16, number of candidates=121,81 and stride=2. The features used are, green, value and hue. Red,blue and gray are expected to show similar results as green channel because the object of interest is white in color.

Channel	numCand	Accuracy	Time(ms/frame)
Red	121	0.45	3.34
	81	0.41	2.63
Green	121	0.50	4.53
	81	0.40	2.42
Blue	121	0.44	4.53
	81	0.42	**2.58**
Gray	121	0.49	6.42
	81	0.45	8.57
Hue	121	0.30	8.84
	81	0.27	5.00
Saturation	121	0.27	9.82
	81	0.24	7.05
Value	121	**0.51**	11.57
	81	0.46	7.62

TABLE III
COLOR BASED TRACKING RESULTS FOR SPHERE. NUMCAND: NUMBER OF CANDIDATES

a) Green Channel: Using the green channel, decent results are observed for the videos. Figure 9 shows the performance of the algorithms with 121 bins. The average processing time is 6.3 ms/frame and the average tracking performance is 50.2%. Implementing the algorithm with 81 candidates, the average processing time is 5.3 ms/frame and the average tracking performance is 39.9%. This is expected because 121 candidates imply larger number of histogram generation and comparison, and hence the time required is slightly larger. However, the accuracy is also increased by 10%. Figure 10 shows the comparison of similarity maps for the two cases. A slightly better graph is observed for 121 candidates with higher values in the later frames in comparison to the one with 81 candidates. As seen in Table VII, the red, blue and gray channel show accuracy between 40% and 45% accuracy for 81 candidates and between 44% and 49% accuracy for 121 candidates which is slightly lower than the ones from green channel. The lower value in blue channel can be attributed to the blue background in the sequence. Similar problem due to background is seen in gray channel. The features will have high values for both ball and background resulting in certain inaccuracy. This is visualised

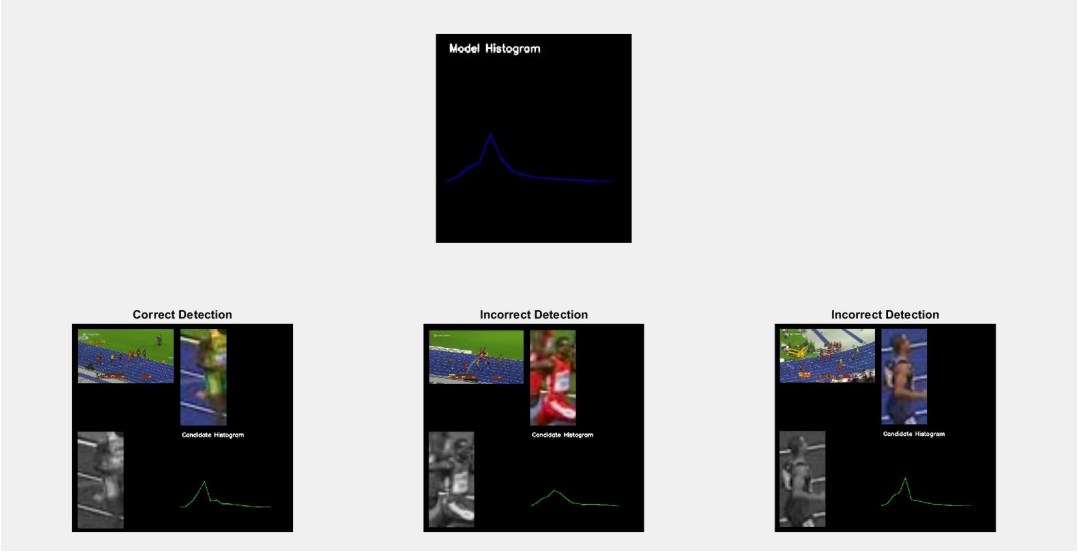

Fig. 7. Results of the track analysis from gray for different frames (bolt1, from table 2).

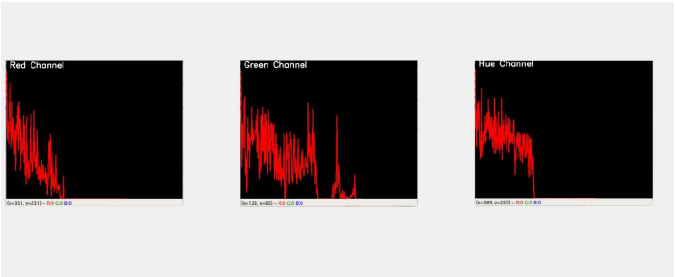

Fig. 8. Similarity plot (bolt1, from table 2) Left: Red, Middle: Green, Right:Hue—Peaks correspond to higher similarity to the ground truth

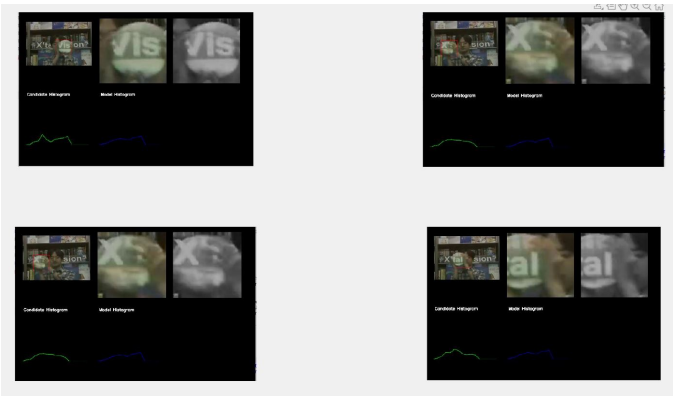

Fig. 9. Results of the track analysis from green channel for different frames (sphere, from table 2).

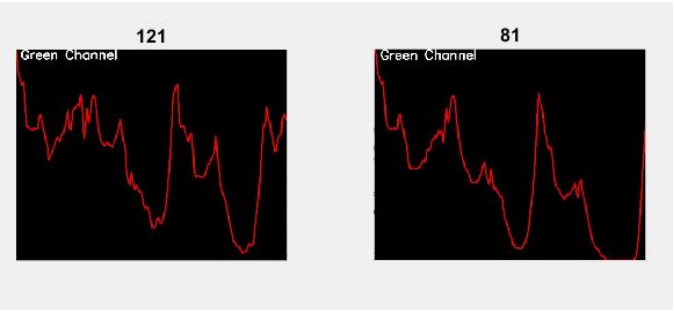

Fig. 10. Similarity plot:Green channel for 121 and 81 candidates (sphere, from table 2)—Peaks correspond to higher similarity to the ground truth

in Figure 11 where the candidate region has the background (containing white text) along with a part of the sphere.

b) Value Channel: Value channel of HSV describes the lightness in an image. Minimum value gives black color and maximum value gives white color. Since the sphere in the sequence has whitish color, value is expected to form a discriminating feature for tracking. Hue and saturation gives the color and its amount. The object of interest is white, therefore they do not form discriminating features. The accuracy obtained are around 30% for both 81 and 121 candidates for these channels. Therefore, further description of these two features are not reported. Figure12 shows the detections for various frame with 121 candidates. For image 1, it is seen that the histogram is the closest to the the model histogram. Image 2 also detects the ball. However, in image 3 and image 4, the detection contains a significant part of the background containing the printed text on the class. A probable reason for this is that the model histogram also contains the text. Therefore, while searching through the neighbourhood, the candidate containing the text also gives a maximum match. Another reason for this could be the low frame rate and slow update of positions which leads to the sphere not being a candidate and the area with the text thus being chosen as a detection. It can be observed for Table VII that the highest score of 51% is obtained for this configuration. With 81 candidates, the accuracy comes down to 46%. Figure 13 shows the comparison between the similarity maps generated by 121 candidates and 81 candidates.

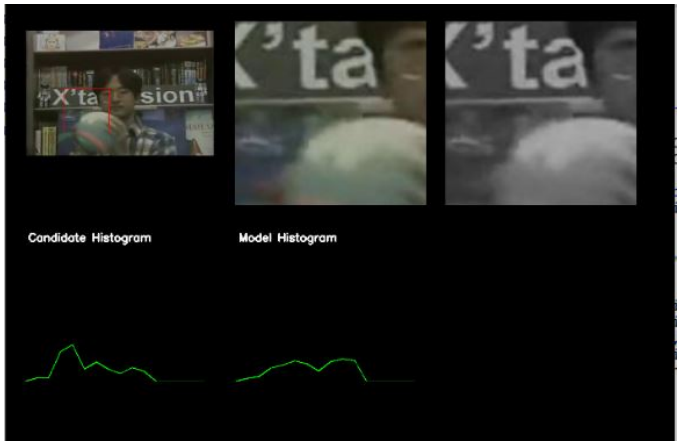

Fig. 11. Results of the track analysis for gray showing inaccurate candidate because of similar background (sphere, from table 2).

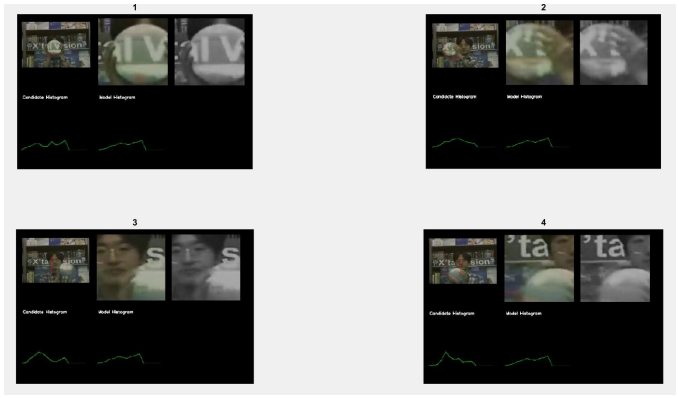

Fig. 12. Results of the track analysis from value channel for different frames (sphere, from Table 2).

From Table VII, the highest score is obtained for the value channel. However, with green channel, the accuracy values obtained are only less by 1% and the computation time is 4.52 ms/frame which is significantly less than the computation time for the value channel (11.57 ms/frame). Therefore, it is a better choice to use the green channel with 16 bins and 121 candidates for object tracking in this sequence.

3) ***Car1****:* In this video the camera is shaking constantly and there are nearby vehicles possessing similar appearance. Furthermore, the scene is not illuminated by white light

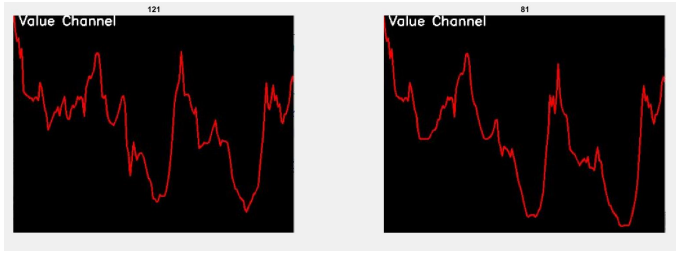

Fig. 13. Similarity plot for value channel for 121 and 81 candidates (sphere, from table 2—Peaks correspond to higher similarity to the ground truth

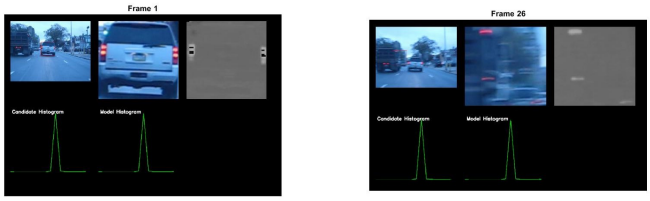

Fig. 14. Results of the track analysis for hue channel for number of frames=121 (car1, from Table 2) image - Top-left:Current Frame, Top-right: Candidate Region, Bottom-Left: Feature, Bottom-right: Histogram of Candidate Region

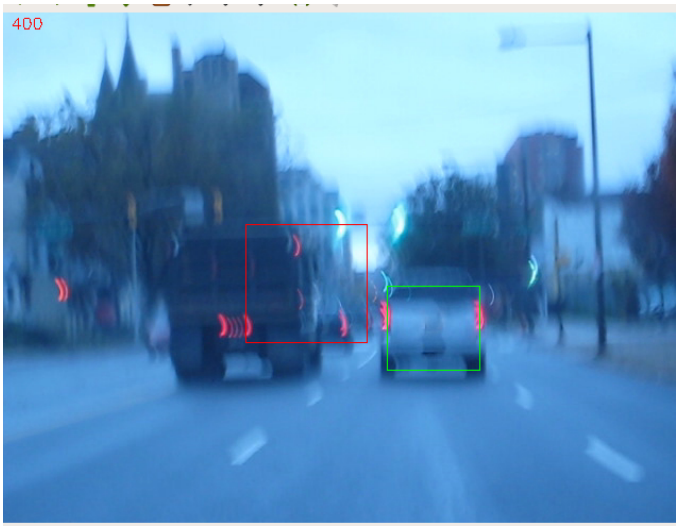

Fig. 15. Results of the track analysis for value channel for number of frames=121 (car1, from Table 2)

and looks to have a visibly blue tint. This video is evaluated for the suggested parameters of bins=16, number of candidates=121,81 and stride=2. The features used are- red and saturation. Blue channel doesn't show discriminative features because of the non-white illumination. Green and gray channels are expected to show similar results as the red channel. As seen in Table IV, hue does not give results as bounding box drifts out of the frame and loses the object early on in the video. This is due to the fact the the sequence is not at all discriminable in the hue channel because of the lack of distinctive colors (except red which is not a sufficient feature for hue). In figure 14, the first image shows model features and the second image shows almost the same features for a completely different candidate. Value also doesn't seem have a higher accuracy because the whole scene is bluish-white and hence hard to discriminate in this channel. Figure 15 shows a frame where the bounding box drifts afar because the detection shows a highly similar histogram even for a wrong candidate and also takes a longer time to process.

a) Red Channel: The red channel is expected to perform well because the object of interest is composed of bluish white, red and black colors that would make it distinguishable. However, other vehicles on the road also have similar red lights which is a problem. Since each of

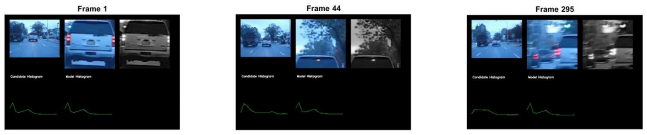

Fig. 16. Results of the track analysis in red channel showing different frames (car1, from table 2). For each image - Top-left:Current Frame, Top-right: Candidate Region, Bottom-Left: Feature, Bottom-right: Histogram of Candidate Region

the frame has a considerable area of the sky and similar to red regions, high values are also assigned to candidates near the white sky. The first image in Figure 17 shows the model candidate which shows white regions for car body and lights and black region for glass areas. In the second image, the red channel shows incorrect detection in frame 44 because the sky contributes to the white region and the leaves contribute to the black region making it look similar to the model. To discriminate the leaves, a similar analysis was carried out in green channel. Since the colors are not very distinctive, the green channel also showed incorrect detections. At frame 295, when the street light is visible, the frame drifts towards the black truck with a similar red light and sky in background. This can be seen in third image in Figure 17 where there is drift in the bounding box region towards the truck. Next section discusses how saturation channel can partly solve this problem. The average processing time is 8.06 ms/frame and average tracking performance is 51% for 121 candidates. Implementing the algorithm with 81 candidates, the average processing time is 5.23 ms/frame and average tracking performance is 44%. As the camera motion is not drastic, it was expected that the drift would be less if the search neighbourhood was smaller. However, with 64 candidates, the performance drops to 33 %. This is because when the bounding box drifts due to camera shaking, the recovery towards object of interest becomes more difficult as the neighbourhood is smaller.

b) Saturation Channel: The issue discussed in the red channel regarding the drift towards street light is better dealt with in saturation channel. This is because the white color of the car and of the street light differ significantly in the saturation values. The first image Figure 14 shows the model candidate where the black glass has lighter values and the white body has darker values. Red lights are not as differentiable in this channel. For frame 295 and 390, it is seen that the bounding box continues to stay around the object of interest because the street light has higher saturation values than the white color of the car thus solving the drifting problem. Further in frame 389, there is another drift shown, that shows the inability of this channel to differentiate between single and multiple cars. The average processing time is 18.18 ms/frame and the average tracking performance is 49% for 121 candidates. The average processing time is 12.30 ms/frame and average tracking performance is 47% for 81 candidates. There is not much difference observed on changing the number of candidates.

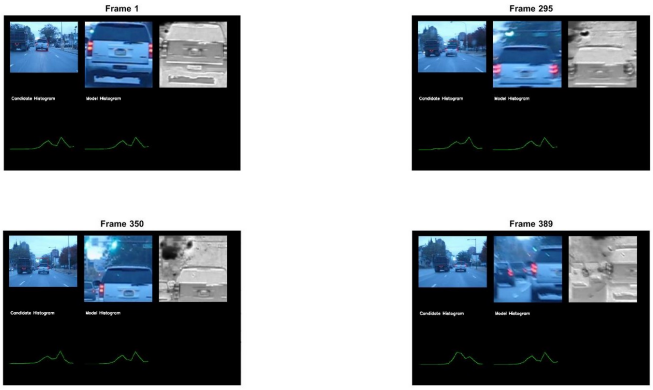

Fig. 17. Results of the track analysis in saturation channel showing different frames (car1, from table 2). For each image - Top-left:Current Frame, Top-right: Candidate Region, Bottom-Left: Feature, Bottom-right: Histogram of Candidate Region

Channel	numCand	Accuracy	Time(ms/frame)
Red	121	**0.51**	8.06
	81	0.44	5.23
Green	121	0.48	7.22
	81	0.45	**5.08**
Blue	121	0.24	7.00
	81	0.36	5.47
Gray	121	0.49	8.09
	81	0.41	5.88
Hue	121	no res	no res
	81	no res	no res
Saturation	121	0.49	18.18
	81	0.47	12.30
Value	121	0.33	19.28
	81	0.38	13.09

TABLE IV
COLOR BASED TRACKING RESULTS FOR CAR1. NUMCAND: NUMBER OF CANDIDATES

The red and saturation channel give comparable accuracy for 121 candidates. However, red channel(8.06 ms/frame) takes lesser time for tracking than saturation channel(18.18 ms/frame). Therefore, the best combination for this video is red channel with 121 candidates, 16 bins and stride 2.

B. *Gradient based tracking*

Gradient based tracking has been tested on bolt1 video and the analysis has been done for the two real video sequences containing different types of tracking challenges. The effect of changing the bins and number of candidates for each of the video sequences had been presented in detail in this section. For each of the sequences stride was set to 2.

*1) **bolt1**:* The video consists of a running race and is used to test the correctness of the implementation. The major challenges for the gradient based tracker is that all the runners are physically similar and have similar pose variations as the race progresses. Initially the tracker works well for upto 160 frames, but nearing a curve in the camera angle the tracker drifts to a nearby runner who looks much similar to Usain Bolt (the person being tracked) in terms of

the gradient features. The results for the video are presented

Fig. 18. Gradient based tracking result of bolt1 for bin=6. Left to right: numCand=81 frame_lost=163, numCand=100 frame_lost=135, numCand=121 frame_lost=46. frame_lost: frame when tracker loses the object

in Table V. The best results are gotten for the default testing parameters bins=9 and number of candidates=100 for the task till frame 150. After that it drifts off to a nearby runner. From Table V it can be seen that for bins=6 the accuracy decreases with increasing the number of candidates. This is because a lower number of bins gives similar descriptors for the different runners. So by increasing the area of search the tracker drifts very easily to the nearby runner and is unable to discriminate between Bolt and another runner as shown in Fig 18.

Bins	NumCand	Accuracy (%)	Time (ms/frame)
6	81	14.27	20.1
	100	12.9	19.36
	121	6.2	23.1
9	81	13.72	**16**
	100	**16.54**	24.3
	121	16.08	20.1
12	81	15.65	16.83
	100	8.28	21.5
	121	15.60	26.8

TABLE V

GRADIENT BASED TRACKING RESULTS FOR BOLT1. NUMCAND = NUMBER OF CANDIDATES

For the case of bins= 9/12 a particular trend cannot be observed. Rather it is evident that both the bins and number of candidates are essential and affect the tracker performance differently for each combination. For bins=6/12, 100 candidates performs worse than 81 candidates but for bins=9 it gives the best performance. This variability is because the descriptors for all the runners are similar. For bins=12, 100 candidates loses the object and is not able to re-align back. In contrast, for bins=9/12, for the candidates=121, the accuracy is better than in the case of bins=6. This is because the tracker drifts away and then is able to realigns with Bolt's legs towards the end of the video. This behaviour is shown by the example detection and the peaks in the similarity graph in Fig 19 towards the end of the video. The larger candidates generated help the tracker to realign when the runner and Bolt come closer in the video.

*2) **ball2**:* As summarized in Section IV, one of the key challenges for the white football is its similarity in colour to the background when the video begins. This challenge need not be considered here, as the gradient based features computed by HOG descriptors does not depend on the color of the object. A few combinations of bins and number of

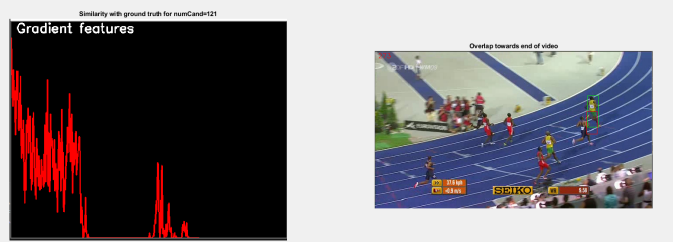

Fig. 19. Gradient based tracking result for bolt1 on the similarity observed at end of video and overlap of the detection of other runner with ground truth. Peaks correspond to higher similarity to the ground truth

candidates have been tested for the video. The results for the combinations of bins and candidates while keeping the stride fixed at 2 are given in Table V-B.2. It can be seen that increasing the number of candidates improves the accuracy of the tracking. Changing the number of bins has almost no effect on the accuracy of the tracker and hence further experiments were performed by changing the number of candidates that are to be considered.

Bins	NumCand	Accuracy (%)	Time (ms/frame)
6	81	6.04	**15.3**
	100	6.05	19.6
	121	6.08	24.5
9	81	6.04	16.2
	100	6.05	21.6
	121	6.13	26.1
12	81	6.05	19.1
	100	6.05	20.1
	121	**6.13**	28.1

TABLE VI

GRADIENT BASED RESULTS FOR BALL2. STRIDE = 2, NUMCAND = NUMBER OF CANDIDATES

On further testing it was observed that a major problem with tracking is that the ball has a high velocity and goes from the top of the frame to the bottom in a very short time (total 41 frames). This causes the tracker to fail right at the beginning even with the 9 bins and 121 candidates and the tracking performance is very poor at approximately 6.1% accuracy.

As the ball is moving is faster and covers more distance from one frame to the other it is necessary to search in a larger area around the previous detection to actually find the ball. Based on this argument a larger number of candidates were tested for the values from $\{121, 196, 289\}$. Using a neighbourhood larger than 19x19 (289) increased the computation time drastically and also many of the generated candidates were outside the image boundary. Changing the value of the stride was also tested as for a smaller number of candidates it would also work in increasing the neighbourhood from which the candidates are generated. The effect of stride and changing the number of candidates on the tracking performance has been shown in Figure 20 and the accuracies have been mentioned in Table VII.

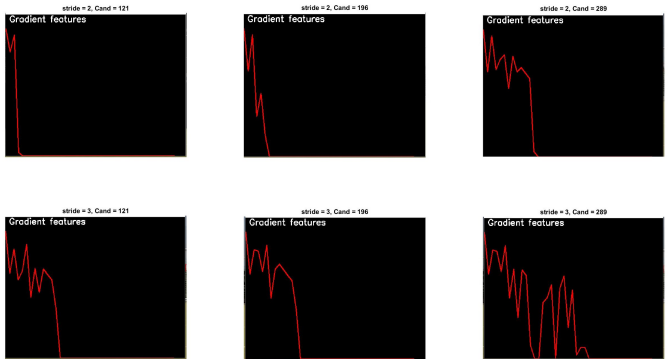

Fig. 20. Results for tracking analysis of gradient features for ball2. Peaks correspond to higher similarity with ground truth

Stride	numCand	Accuracy (%)	Time (ms/frame)
2	121	6.1	**24.75**
	196	7.8	38.21
	289	**19.5**	57.33
3	121	19.3	**23.86**
	196	20.3	37.86
	289	**25.6**	56.86

TABLE VII
GRADIENT BASED TRACKING RESULTS FOR BALL2. NUMCAND: NUMBER OF CANDIDATES

It is evident that for the same number of candidates increasing the stride is helpful while keeping the computation similar. As it can be seen the tracker with stride=3 and 289 candidates performs the best from the given combinations. It loses the football on the way when it nears the net but is able to re-align back and loses it completely only after the ball touches the ground. Figure 21 shows the frames at which the tracker lost the object for the case of changing the candidates in $\{121, 289\}$ and the stride taken from $\{2, 3\}$.

*3) **basketball**:* This video comprises a short sequence from a basketball game where the object to be tracked is a player from the team wearing the green jersey. The opposite team comprises of players in white jersey and also the referee of the game can be seen in the frames towards the second

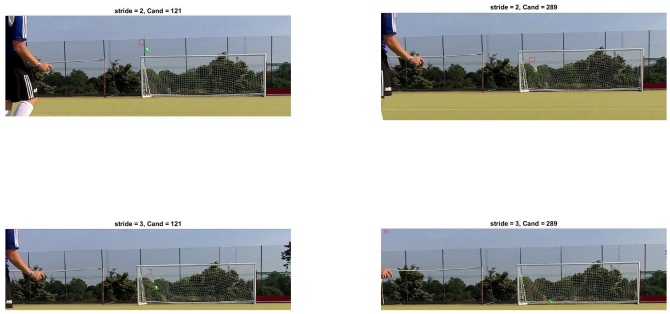

Fig. 21. Comparison for different parameters and the frames where they lose the object. For stride=2, candidate=121 frame_lost = 7, stride=2, candidate=289 frame_lost = 17, stride=3, candidate=121 frame_lost = 18 and stride=3, candidate=289 frame_lost = 32

half of the video. For gradient based descriptors the players all have similar poses and hence drifting of the tracker to another player was one of the main issues. In addition, the players are moving fast, so using a larger neighbourhood generally worked well as seen in Table VIII.

Bins	NumCand	Accuracy (%)	Time (ms/frame)
6	81	9.78	**15.8**
	100	25.4	19.3
	121	**55.22**	23.6
9	81	24.32	16.6
	100	24.92	20.3
	121	25.50	24.8
12	81	9.3	17.1
	100	24.73	21.3
	121	10.18	25.67
7	**121**	**57.68**	**23.2**

TABLE VIII
GRADIENT BASED TRACKING RESULTS FOR BASKETBALL. NUMCAND=NUMBER OF CANDIDATES

The effect of choosing the correct number of bins for the descriptor is important for this video as the descriptors need to be able to discriminate between different players who look similar and also as the person being tracked has a changing posture. It is observed from Table VIII that increasing the bins worsens the tracker performance. In the case of 12 bins,

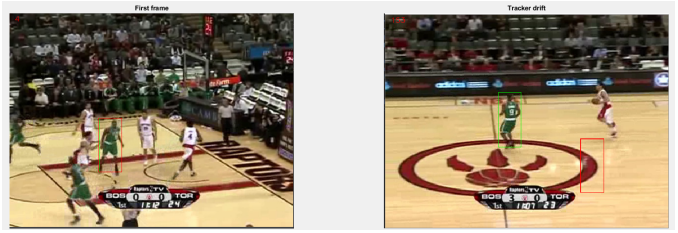

Fig. 22. Gradient based tracking of bins=12, number of candidates = 121, drift due to background symbol of court

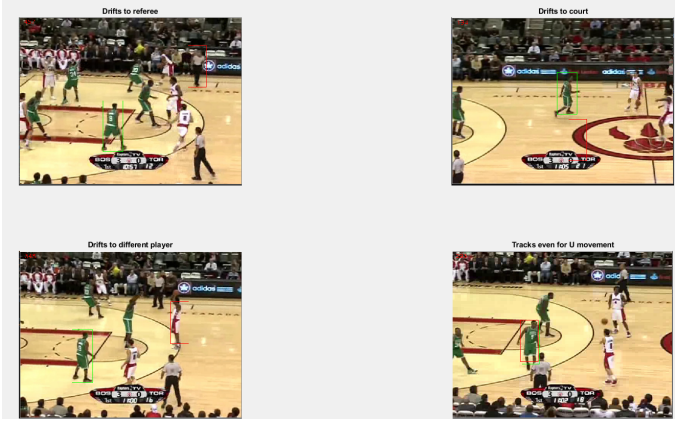

Fig. 23. Results for gradient based tracking. First three show the different errors made by the low accuracy trackers. Last image shows the best tracker performing correctly even for the end of the video

using 121 candidates gives a worse performance than the

100 candidates case. The tracker drifts to the symbols on the basketball court when the tracked player runs. This is because the model histogram contains some details of the symbol which is present in the first frame as the background of the player. This issue is shown in Fig 22. Based on this behaviour it can be concluded that having lower number of bins reduces the impact of the background. Most of the combinations lose the object and drift off to the nearby players or the referee when they lose track of the object as shown in Fig 23. These people are similar in pose to the tracked player and so this drifting occurs. The best combination is observed with bins=7 and number of candidates=121. The tracker is able to follow the player when he runs and also when he moves in U near the basket as can be seen in Fig 23. The other combinations except bins=6 and number of candidates=121 lose the player after he crosses halfway across the court. This can be seen in Fig 24 where the peaks are only till halfway for the lower accuracy trackers and the best combination keeps getting peaks of matches till the end of video. For the sake of representation, the graph of only one lower accuracy tracker is shown. The others have results similar to it.

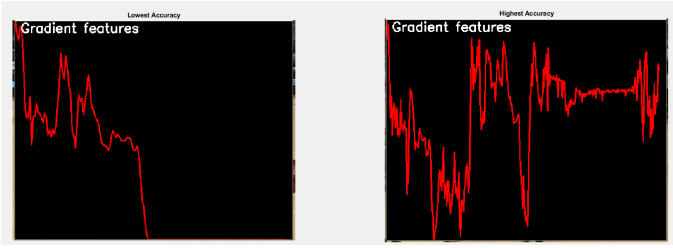

Fig. 24. Similarity for gradient based tracking for basketball for the best and lowest accuracy trackers from Table VIII. Peaks correspond to higher similarity to the ground truth

C. *Fusion based tracking*

The analysis of the fusion based tracking has been performed for four sequences. The fusion based tracking offers eight features for color based and gradient based tracking. The sequences are tested by color model(by choosing best feature), gradient model and fusion of the two models. The performance is evaluated for these models and is listed in Table IX. The following parameters are used for testing, bins=16, candidates=225 and stride=2. Higher values of bins and candidates do not improve the accuracy and increases the computation time.

1) ***bolt1****:* As discussed in previous sections, color based model gives best result for the green feature. The average processing time is 8.03 ms/frame and average tracking performance is 26.4% for 225 candidates. Furthermore, the gradient model fails to track well due to similarity between the object of interest and the nearby objects. It only achieves a 4.5% accuracy with time 78.33 % ms/frame. Figure 25 shows the similarity plots for each of the models for this sequence. Therefore, the fusion of the model for the color and gradient based trackers given these parameters does not give significant improvement in the results. It is advantageous to use color based model as it has a better performance and saves computation time.

Seq	Model	Accuracy(%)	Time(ms/frame)
bolt1	**Green**	**26.4**	**8.03**
	Gradient	4.5	78.33
	Fusion	26.2	72.43
ball	**Red**	**65.6**	**7.59**
	Gradient	11.4	81.21
	Fusion	63.5	82.33
bag	**Gray**	**37.5**	**27.50**
	Gradient	4.7	80.97
	Fusion	37.4	84.89
road	Value	31.7	**16.19**
	Gradient	**48.8**	64.85
	Fusion	30.1	76.78

TABLE IX
FUSION BASED TRACKING RESULTS

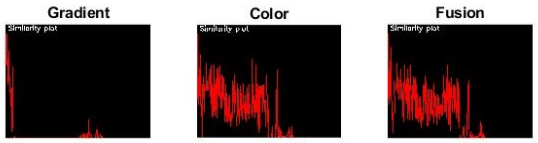

Fig. 25. Similarity plot of Color, Gradient and fusion based tracking (bolt1, from Table 2)—Peaks correspond to higher similarity with ground truth

2) ***ball****:* This sequence shows a red and white football being kicked back and forth. Red channel gives the best accuracy values because the feature gives high values for both red and white color. The average processing time is 7.59 ms/frame and the average tracking performance is 65.6% for 225 candidates. With variable bounding box size, this accuracy could be increased because the size of ball changes throughout the sequence. Figure 26 shows the oscillation in the similarity values across the frames for the color model as the object decreases in size while moving away from the camera and then regains original size (as in the model) as it comes closer to the camera. This is the reason for the poor performance of the gradient model with just 11.4% accuracy. The fusion model gives 63.5 % accuracy and takes a higher computation time. Therefore, it is advantageous to use only the color based model.

3) ***bag****:* This sequence shows a white bag floating in air. There are frequent changes in the distance between camera and the bag. The bag also deforms and changes in appearance

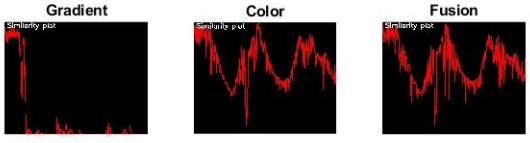

Fig. 26. Similarity plot for Color, Gradient and Fusion based tracking (ball, from Table 2)—Peaks correspond to higher similarity with ground truth

Fig. 27. Similarity plot of Color, Gradient and Fusion based tracking (bag, from Table 2)—Peaks correspond to higher similarity with ground truth

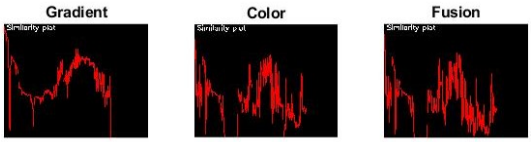

Fig. 28. Similarity plot of Color, Gradient and Fusion based tracking (road, from Table 2)

throughout the video. It is seen that gray channel works best for this sequence with an average tracking performance 37.5% and time taken as 27.5 ms/frame. Figure 27 shows a dip in similarity values across the frames in the color model as the bag decreases in size in the middle of the sequence but the size of detection's bounding box remains the same. Later, when the object regains original size (as in the model) a peak in the similarity is observed. Gradient based model fails for this sequence because of the constant deformation of object and so gives only a 4.7% accuracy. The fusion model gives 37.4 % accuracy with higher computation time. It is advantageous to use color based model.

*4) **road**:* This sequence shows a biker(white) riding on the road. It has frames with lower resolution and highly saturated color values. The background is mostly green in color and the biker follows wavy trajectory. Blue channel does well in identifying the biker with 32 % accuracy. However, this section demonstrates the potential of the fusion model by using the value channel. It gives an average tracking performance 31.7% and average processing time 16.19 %. The gradient model gives highest accuracy 48.8% with 64.25 ms/frame processing time. Figure 28 demonstrates the same with plots assessing the similarity with ground truth for each of the models. Figure 29 shows that at frame 390, the gradient model fails to identify the object due to appearance change. Since value-based model can identify the biker, a better detection is obtained in fusion based tracking result. However the average accuracy remains low(30%). So it is advantageous to use gradient based model.

VI. CONCLUSION

In conclusion, the current implementation of color based, gradient based and fusion based tracking is used to track a single object in a sequence. For the color based tracker, the feature to be used is tailored to the sequence and the object of interest. Gray and RGB channels take significantly less computation time. However, HS features have advantages, such as robustness to illumination change, which prove beneficial for certain situations. It was observed that having more than 100 candidates made the algorithm more robust to incorrect detections. For the gradient based tracking it is seen that it is invariant to color changes. But when there are many similar objects in the scene then the parameters of number of bins and candidates need to be tuned well for the tracker to give a good performance. Lastly, for the fusion based model, since scores for both the models are added, fusion helps to recover in case one of the model fails to identify the object of interest. However, for the given sequences, it did not improve the accuracy scores and it also worsened the performance in some frames. The combination gave bad results when one of the trackers performed significantly worse than the other. In such a case not using a fusion approach would be beneficial.

Fig. 29. Comparison of Color, Gradient and Fusion based tracking for frame 390 (road, from Table 2)

As a future work, the model scores could be combined using weighted sum or the model giving higher score can be chosen at run time to give the detection for the given frame of the video.

VII. TIME LOG

- **Color based tracking**: 3.5 hours: 1 hour to set up the project and reading the slides. 2.5 hours for the implementation and testing using 'bolt1' video.
- **Gradient based tracking** 2.5 hours: 0.5 hour to read the paper, understand the gradient based approach method. 2 hours to code and test.
- **Fusion of colour and gradient**: 2.5 hours: 2 hour to implement the combination and set up the three different modes of operation (color, gradient or fusion) in the code. 0.5 hours debugging
- **Optimization**: 2.5 hours: Spent in structuring the classes, removing unnecessary functions and variables.
- **Evaluation**: 4 hours: For each video trying and checking what parameters(feature, stride, bins and number of candidates) work best for each of them.
- **Report** 10 hours: Writing, generating screenshots for figures, comparing on other dataset, proof-reading, editing and adding references.

REFERENCES

[1] Birchfield, S., 1998, June. Elliptical head tracking using intensity gradients and color histograms. In Proceedings. 1998 IEEE Computer Society conference on computer vision and pattern recognition (Cat. No. 98CB36231) (pp. 232-237). IEEE.

[2] Fieguth, P. and Terzopoulos, D., 1997, June. Color-based tracking of heads and other mobile objects at video frame rates. In Proceedings of IEEE computer society conference on computer vision and pattern recognition (pp. 21-27). IEEE.